Enhancing Large Language Models Reasoning via Multi-Path Optimization on Knowledge Graph

Jiyong Liao, Chubo Liu, *Member, IEEE*, Yan Ding, *Member, IEEE*, Haotian Wang, Zhuo Tang, *Member, IEEE*, Kenli Li, *Senior Member, IEEE*, and Keqin Li, *Fellow, IEEE*

***Abstract*—Large language models (LLMs) have demonstrated remarkable reasoning and generation capabilities in various natural language tasks. However, they often struggle with hallucinations or reasoning errors, particularly when handling domain-specific knowledge or complex multi-hop reasoning. The integration of knowledge graph (KG) provides LLMs with structured and reliable contextual knowledge, effectively mitigating issues of factual accuracy and incomplete reasoning chains. Nevertheless, existing KG-guided LLM reasoning methods still face challenges, including narrow answer coverage, limited accuracy in multi-hop reasoning, and inefficiency caused by frequent LLM API calls. To address these problems, we propose ELMK (Enhancing Large language models reasoning via Multi-path optimization on Knowledge graph), a novel KG-based LLM reasoning method that improves output comprehensiveness and interpretability. ELMK follows a retrieval–embedding–reasoning pipeline. First, depth-first search is used to extract relevant reasoning graphs, then a multi-path encoder is trained to semantically encode the question and candidate paths for precise path selection. Subsequently, the multi-path exploration strategy divides paths into multiple semantic clusters and selects the most similar paths from each cluster to ensure diverse coverage and complete answers. Finally, these paths are combined with prompts to guide LLMs toward reliable outputs. Extensive experiments on public benchmarks demonstrate that ELMK outperforms several state-of-the-art methods in terms of performance and generates more faithful and interpretable reasoning results.**

***Index Terms*—Retrieval-augmented generation, knowledge graph, large language model, answer coverage.**

Received 29 September 2025; revised 4 March 2026; accepted 6 April 2026. Date of publication 8 April 2026; date of current version 2 June 2026. This work was supported in part by the Programs of NSFC under Grant 62441234, Grant 62502151, Grant 62532005, and Grant 62472193 and in part by the Natural Science Foundation of Hunan Province under Grant 2024JJ7383. Recommended for acceptance by D. Yang. *(Corresponding author: Chubo Liu.)*

Jiyong Liao is with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China, and also with the School of Computer and Artificial Intelligence, Huaihua University, Huaihua 418000, China (e-mail: liaojiyong@hnu.edu.cn).

Chubo Liu is with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China, and also with the Ministry of Education Key Laboratory of "Fusion Computing of Supercomputing and Artificial Intelligence", Changsha 410082, China (e-mail: liuchubo@hnu.edu.cn).

Yan Ding, Haotian Wang, Zhuo Tang, and Kenli Li are with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China, and also with National Supercomputing Center in Changsha, Changsha 410082, China (e-mail: ding@hnu.edu.cn; wanghaotian@hnu.edu.cn; ztang@hnu.edu.cn; lkl@hnu.edu.cn).

Keqin Li is with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China, also with the National Supercomputing Center in Changsha, Changsha 410082, China, and also with the Department of Computer Science, State University of New York, New Paltz, NY 12561 USA (e-mail: lik@newpaltz.edu).

Our code and data are available at https://github.com/Liaojiyong/ELMK.

Digital Object Identifier 10.1109/TKDE.2026.3682046

I. Introduction

LARGE language models (LLMs) have demonstrated excellent text generation, semantic understanding, and logical reasoning abilities in the field of natural language processing (NLP) through large-scale text training, and are widely used in dialogue systems [1], knowledge graph question answering (KGQA) [2], [3] and other fields [4]. Despite their impressive performance, LLMs still face two critical questions: First, the lack of domain-specific prior knowledge often leads to hallucination [5], [6] — content that seems credible yet contains incorrect statements; second, the substantial expense of training hinders the dynamic updating of model parameters [7], while fine-tuning may trigger catastrophic knowledge forgetting risk. An example is present in Fig. 1(a), models like GPT may generate incorrect answers due to outdated internal knowledge or reasoning hallucination in LLMs. Retrieval-augmented generation (RAG) [8], [9], [10], [11] has been widely adopted as an effective solution to mitigate hallucination. The core idea is to enhance reasoning capabilities of LLMs by retrieving knowledge from external databases that is highly relevant to user queries. However, as illustrated in Fig. 1(b), in multi-hop reasoning tasks, RAG methods are limited to extracting information solely from text segments that include anchor entities. As the retrieval granularity decreases, the obtained knowledge fragments often lack inherent connections, which can easily lead to wrong answers and insufficient interpretability.

Knowledge graph (KG) is considered a promising solution for enhancing LLM reasoning capabilities due to their structured nature [6], [12], [13], [14]. By leveraging explicit entity relationships, KGs provide reliable knowledge support for complex tasks such as multi-hop reasoning. KGQA as a typical task for evaluating the collaborative reasoning of LLMs and KGs, mainly adopts two paradigms: Semantic parsing approaches [15], [16] that convert natural language questions into structured queries, and retrieval-augmented methods [7], [10] that assist LLM reasoning by dynamically retrieving relevant fact triples from KG. These approaches generally follow an iterative reasoning framework of "entity identification–path retrieval–evidence optimization," ultimately generating answers in KG factual evidence through multiple rounds of knowledge retrieval and path optimization.

Although KG has made substantial progress in LLM reasoning, there are still three main challenges.

(i) Low answer coverage (i.e., unable to comprehensively cover multiple correct answers to a question): In Fig. 1(c),

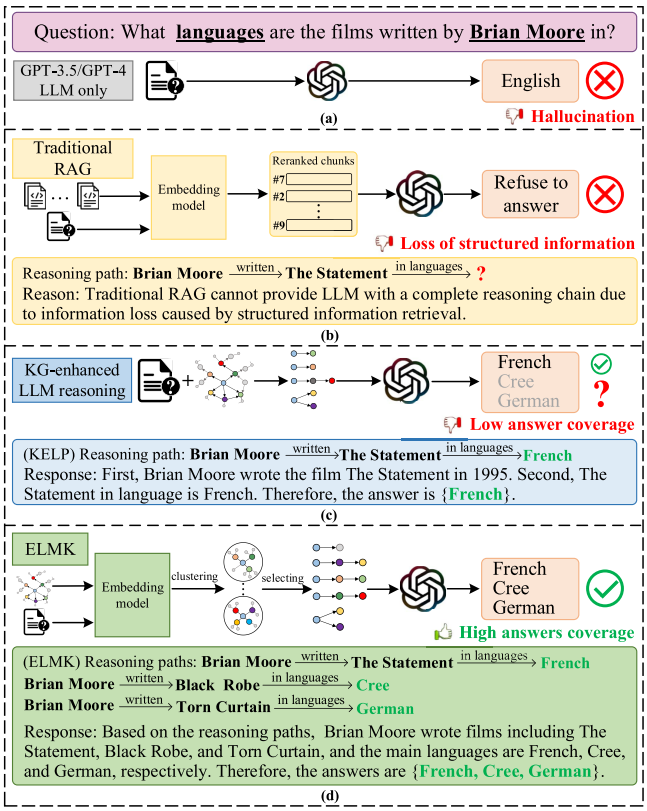

Fig. 1. Comparison between our method and other baselines: (a) LLM-only, (b) traditional RAG, (c) KG-enhanced LLM reasoning, (d) our proposed ELMK. The question is from a 2-hop test data in the MetaQA dataset.

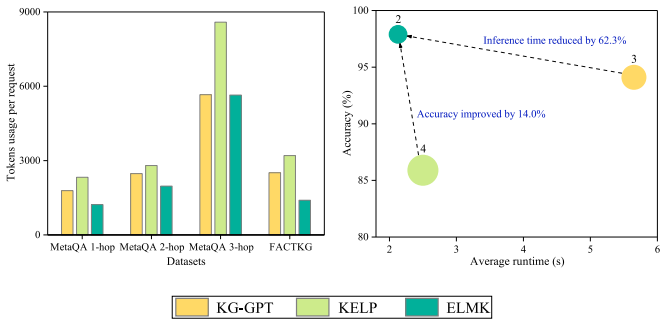

Fig. 2. Comparison of overhead between ELMK and baselines. Token usage (left), detailed performance (right) on MetaQA 2-hop. The number and circle size indicate the number of LLM calls.

existing methods [5], [17], [18] often lack a deep understanding of KG structures when addressing question with multiple correct answers, resulting in constrained reasoning paths: Critical reasoning paths are easily overlooked and the coverage is narrow, making it difficult to encompass all potential correct answers. This significantly impacts the completeness and accuracy of the answers, particularly in decision-support scenarios that require comprehensive consideration of multiple possibilities.

(ii) Frequent API calls can significantly increase system overhead: In Fig. 2, we compare the execution cost of ELMK with state-of-the-art methods. The results show that KELP [17], KG-GPT [18] require more LLM calls and inference time when executing knowledge retrieval and complex reasoning on large-scale datasets. However, for closed-source LLMs like GPT and Claude 4 that are only accessible through paid APIs, this leads to heavy token consumption and significant time costs.

(iii) Low multi-hop reasoning accuracy: During the handling of complex multi-hop tasks [13], [17], [18], LLMs tend to generate intermediate reasoning paths that appear plausible but are actually incorrect, leading to the premature exclusion of potential correct answers. Moreover, as the number of reasoning hops increases, LLMs must process longer dependency chains, which not only requires larger context window but also significantly increases computational overhead.

To address the aforementioned challenges, we propose a novel KG-augmented approach called **E**nhancing **L**arge language models reasoning via **M**ulti-path optimization on **K**nowledge graph (ELMK). As illustrated in Fig. 1(d), this method aims to flexibly select multiple reasoning paths that meet user query requirements, using them as multi-hop chains to enhance the interpretability and completeness of LLM outputs. The ELMK consists of four core modules: (i) Reasoning graph retrieval, (ii) sub-dataset construction and multi-path embedding, (iii) multi-path exploration, and (iv) answer generation. Specifically, the framework first extracts initial paths from the KG as candidate knowledge based on the entity mentioned in the question, constructing the required reasoning graph. Next, a training sub-dataset is constructed, and multi-path encoder is trained to encode both the query and multiple candidate paths within the reasoning graph. This module provides an efficient encoding mechanism, enabling quantitative similarity measurement between the query embedding and candidate path embedding, thus supplying precise semantic information for subsequent path selection. Then, the multi-path exploration strategy is applied to divide the encoded candidate paths into multiple semantic clusters. The top-k reasoning paths are selected from each cluster based on similarity score to ensure the model covers diverse semantic knowledge, thereby improving answer coverage. Finally, the selected relation paths are combined with meticulously crafted prompts and input into the LLM to generate the final response. ELMK can seamlessly integrate various KGs into any LLM without requiring updates to the parameters of LLM. The key contributions of this work are outlined below:

- We propose a novel ELMK method that constructs a path-aligned sub-dataset to pre-train a multi-path encoder for efficient semantic encoding, endowing it with fine-grained semantic discrimination capability.
- We introduce a clustering mechanism to extract semantically diverse relational paths from candidate paths, which suppresses homogeneous noise while improving answer coverage, thereby enhancing the accuracy and completeness of LLM outputs.
- We conduct extensive experiments on KGQA datasets, demonstrating that ELMK outperforms several strong baselines, particularly in dealing with questions with multiple correct answers, fully validating its effectiveness and generalization capability.

TABLE I
COMPARISON OF TRADITIONAL RAG, GRAPHRAG, AND ELMK

		Traditional RAG [8], [9], [11], [18]	**GraphRAG** [5], [13], [14], [17]	**ELMK (ours)**
Key techniques	Retrieving	Breadth-first search or random walk	Nearest neighbor search	Depth-first search
	Pre-training	BERT and its variants (single path)		SentenceBERT (multi-path)
	Pruning	Beam search or LLM agent		Clustering
	Reranking	Required	None	Required
Execution efficiency	Average Runtime	More	Medium $\sim$ More	Less
	Average LLM Calls	High (four or more times)	Medium	Low (two times)
	Average Tokens/request	More (thousands of tokens)	Medium $\sim$ Less (only reasoning)	
Limitations or Advantages		① ②	① ③	④ ⑤

① Poor multi-hop accuracy. ② Structured information missing. ③ Low answer coverage. ④ High answers coverage. ⑤ Low computational cost.

II. RELATED WORK

Our research is closely related to traditional RAG and knowledge graph-enhanced LLM reasoning (GraphRAG). As shown in Table I, we briefly summarize existing related work and compare ELMK with representative methods from various aspects, including the key techniques, execution efficiency, and overall limitations or advantages.

A. Retrieval-Augmented Generation (RAG)

In enhancing the reasoning capabilities of LLMs, retrieval-augmented generation has emerged as a pivotal technique for handling complex tasks. Researchers have explored reasoning design methodologies from multiple dimensions, encompassing zero-shot/few-shot prompting [8], [11], [19] and divide-and-conquer strategy [9], [18] and chain-of-thought reasoning (CoT) [20], [21], [22].

Without relying on model fine-tuning, these methods have notably boosted LLM performance across a wide range of application scenarios and domains. For example, KG-GPT [18] employs a divide-and-conquer approach to decompose complex reasoning tasks into multiple subtask sequences, substantially enhancing the reasoning capabilities of LLMs. Self-RAG [9] combines retrieval and self-reflection mechanisms to enhance the factuality of answers. Meanwhile, the authors first introduced the concept of the CoT [21] in reasoning tasks, which simulates human cognitive processes by incrementally deriving intermediate steps or sub-goals to guide the model toward more accurate conclusions. Building upon this approach, various extensions including Complex-CoT [20] and Pattern-CoT [22] have further enhanced its effectiveness and applicability. However, these approaches predominantly draw on what the LLM already knows internally, often resulting in factual errors when handling knowledge-intensive tasks and domain-specific scenarios. To address this fundamental challenge, the paradigm of knowledge graph augmented LLM reasoning has emerged as a research focus, which significantly improves factual accuracy and logical interpretability of LLMs by injecting structured domain knowledge and constructing explicit reasoning paths.

B. KG-Enhanced LLM Reasoning

Knowledge graphs, as structured external knowledge sources, have become an important approach to elevate LLM reasoning due to their strong interpretability and logical structure. Current research primarily explores two main directions: Joint learning during the training stage and knowledge injection during the inference phase. During the pre-training phase, researchers employ graph neural networks (GNNs) [23], [24] or multi-task learning frameworks [25], [26] to jointly model KG embedding and the semantic representations of language models, capturing complex relationships between nodes for end-to-end knowledge-aware reasoning. Representative works include KEPLER [27] and CoLAKE [28] integrate text corpora and KG information to achieve efficient heterogeneous information fusion. K-BERT [29] injects triples into pre-trained BERT, enabling flexible integration of domain knowledge. Recent studies have explored implicit knowledge injection, such as DKPLM [30] injects triples during pre-training, fine-tuning, and inference to enhance semantic understanding and reasoning capability. ChatKBQA [3] employs fine-tuned LLMs to generate logical expressions and utilizes unsupervised retrieval methods to replace entities and relations for efficient reasoning. However, these methods still face practical challenges including high retraining costs and frequent knowledge updates, limiting their scalability.

In contrast, knowledge injection during the inference stage dynamically incorporates KG information into context of the model without altering its parameters, significantly enhancing reasoning consistency and interpretability. Recently, various KG-augmented approaches such as LightPROF [6], ToG [13], PoG [14], and RoG [31] have been proposed. The key idea of these methods is to leverage top-k retrieval to extract relevant reasoning paths from KG and feed them together with query into the LLM to generate more accurate answers. Unlike these methods, ORT [32] introduces a reverse thinking approach that traces back from the target to known conditions to guide the LLM in answer generation. HiRAG [10] incorporates knowledge into the KG indexing and retrieval stages through hierarchical indexing and retrieval mechanisms, thereby enhancing semantic relevance while alleviating the knowledge hierarchy gap. KELP [17] further trains an encoder to encode single path as a quantitative basis for the model to select valuable paths. Despite avoiding the high expense of fine-tuning, such methods typically underexploit knowledge graph structures, resulting in incomplete reasoning paths (i.e., low coverage). Particularly for questions with multiple correct answers, where critical information is prone to be omitted, ultimately compromising the completeness and accuracy of the responses.

TABLE II
NOTATIONS AND DESCRIPTION

Notations	Description
D_{sub}	The constructed sub-dataset.
q	The natural language question.
a	The ground-truth answer to question q.
c	The number of clusters.
k	The number of reasoning paths.
p^+, p^-	The positive and negative samples, respectively.
z_q, $z_{p_i^+}$, z_{p^-}	The vector representations of the question q, the i-th positive sample, and the negative sample encoded by the initial encoder, respectively.
$\text{sim}(\cdot,\cdot)$	The cosine similarity function.
C_i	The i-th semantic cluster.
$\mathcal{L}_{\text{const}}, \mathcal{L}_{\text{BCE}}$	Contrastive loss and binary cross-entropy loss.
α	The loss function weight coefficient.
Encoder_θ	Multi-path coding model.
$\mathcal{P}_t$	The final selected reasoning path set.

III. PRELIMINARY

Knowledge Graph (KG) is a data structure that can be represented as $\mathcal{G} = \{\mathcal{E}, \mathcal{R}, \mathcal{T}\}$, where $\mathcal{E}$ and $\mathcal{R}$ denote the set of entities and relations, respectively. $\mathcal{T} = \{(e_h, r, e_t) \mid e_h, e_t \in \mathcal{E},\ r \in \mathcal{R}\}$ is a set of relational facts in form of triples. Here, e_h, r and e_t represent the head entity, relation, and tail entity.

Relation Links are defined as a sequence of relations $z = \{r_1, r_2, \ldots, r_L\}$, where $r_i \in R$ denotes the i-th relation in the link, and L represents the number of hops (or length) for relational reasoning.

Multi-hop Reasoning Paths can be regarded as a series of triple sequences in knowledge graph: $Path = \{(e_{i-1}, r_i, e_i)\}_{i=1}^{L}$, where $e_i \in \mathcal{E}$ denotes the i-th intermediate entity in the path, r_i represents the i-th relation in the relation links. Such a path enables the system to uncover hidden connections between seemingly unrelated entities, thereby expanding the knowledge boundary and providing more accurate and interpretable answers.

Example: Consider a multi-hop reasoning path between "Andrew" and "Harvard University" in knowledge graph: $Path =$ {(Andrew, has child, Jane Smith), (Jane Smith, graduated from, Harvard University)}, and can be visualized as:

$$\textit{Andrew} \xrightarrow{\textit{has child}} \textit{Jane Smith} \xrightarrow{\textit{graduated from}} \textit{Harvard University.}$$

Therefore, based on this reasoning path, we can infer that Andrew's child graduated from Harvard University. The path length is 2.

Knowledge Graph Question Answering (KGQA): is a reasoning task that leverages KG structure to automatically answer questions posed in natural language. The core lies in mapping the input question q to the entity set $\mathcal{E}$ and relation set $\mathcal{R}$ in the KG $\mathcal{G}$, and then finding the correct answer entity $a \in \mathcal{E}$ via multi-hop reasoning paths $Path$, such that $a \in \text{Answer}(q, G)$.

In Table II, we define the notations related to ELMK and their corresponding meanings used in this paper.

IV. METHODOLOGY

A. Overview

The overall framework of ELMK, as illustrated in Fig. 3, consists of four main components: (i) Reasoning graph retrieval, we initially employ a depth-first search (DFS) to search for candidate relationship paths with h hops, starting from question-linked entities, to construct the reasoning graph. (ii) Sub-dataset construction and multi-path embedding, leveraging the original KGQA dataset to generate positive and negative samples for constructing the sub-dataset, followed by training a multi-path encoder to encode both the question and candidate paths, and calculating their similarity to provide precise semantic information for subsequent path selection. (iii) Multi-path exploration, employing a clustering algorithm to divide encoded candidate paths into multiple semantic clusters, selecting the top-k reasoning paths from each cluster based on similarity score to ensure the model covers diverse semantic knowledge paths. (iv) Answer generation, combining a thoughtfully designed prompt template with the selected reasoning paths to construct prompts, then input them into LLM to perform reasoning and produce interpretable outputs.

B. Reasoning Graph Retrieval

In this section, we introduce the reasoning graph retrieval method of ELMK, which aims to identify potentially valuable candidate reasoning paths from the KG. Specifically, starting from the topic entity e of the question q, we employ a DFS strategy to expand the neighboring entities and their associated triples within the multi-hop reasoning range, thereby integrating indirect yet reliable path information related to the query into the reasoning graph $\mathcal{G}_q$, which is composed by merging multiple multi-hop reasoning paths. In this process, the entity set $\mathcal{E}_q$ is updated using newly introduced intermediate entities, which act as bridging subject entities. Here, $\mathcal{E}_q$ includes the topic entity $T(q)$ as well as the intermediate entity set $\{(e, L) \mid e \in T(q)\}$, ensuring that all pertinent entities and reasoning paths are connected within the specified number of hops. These paths $Path$ in the reasoning graph $\mathcal{G}_q$ will serve as candidate samples for the subsequent encoding phase.

C. Sub-Dataset Construction and Multi-Path Embedding

For complex multi-hop KGQA tasks, there are often involve multiple correct answers. To effectively encode multiple reasoning paths simultaneously and improve answer accuracy, we divide the encoding process into two main stages: Sub-dataset construction and multi-path embedding.

1) Sub-Dataset Construction: Given a question q, to enable the encoder to more accurately encode multiple candidate paths, we construct a dedicated sub-dataset from the original KGQA dataset for training the knowledge encoder. This dataset includes question, ground-truth answers, a set of positive samples, and a single negative sample. The specific construction process is as follows: We sample one instance from every five examples as training data. For each given question, if the LLM can directly generate the correct answer, the question is skipped. Otherwise, multiple reasoning paths provided by the KG are used

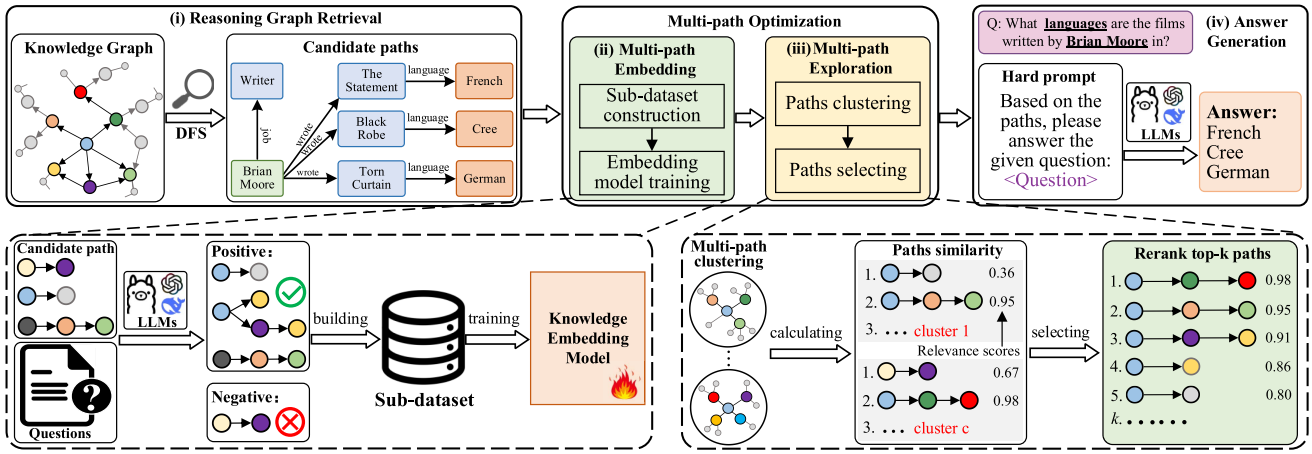

Fig. 3. Overview of the ELMK architecture. (i) Reasoning graph retrieval: Generate candidate paths from the KG to construct reasoning graph; (ii) Sub-dataset construction and multi-path embedding: Build a sub-dataset and pre-train an encoder to encode queries and candidate paths; (iii) Multi-path exploration: Use a clustering algorithm to group paths into semantic clusters, and select the top-k paths from clusters as the final reasoning paths based on similarity score, (iv) Answer generation: Refined path evidence plus a deterministic hard prompt enables dependable, interpretable reasoning.

as the input context for the LLM to guide answer generation. If the predicted answer exactly matches the ground-truth answer, these corresponding reasoning paths (i.e., the sets of triples) are considered a set of positive samples. Conversely, if the LLM generates an incorrect answer, the erroneous answer is used as the tail entity of the triple, and a relation that differs from those in the positive sample set is randomly selected to form a negative sample for training. The sub-dataset $\mathcal{D}_{\text{sub}}$ can be formulated as:

$$\mathcal{D}_{\text{sub}} = (q, a, \text{set}(p^+), p^-) \tag{1}$$

where q denotes the question, a is the corresponding ground-truth answer, set(p^+) represents the set of positive samples, and p^- signifies the negative sample.

To enhance the comprehensiveness and accuracy of answer generation, each question q encompasses a wide range of candidate paths, including knowledge directly related to the answer, as well as paths that are not directly semantically associated but still hold potential influence. Through this design, the encoder can learn deeper latent semantic representations, capturing not only direct semantic connections but also effectively utilizing indirect paths that play a crucial role in the reasoning results. This strategy effectively improves the generalization ability and reasoning performance of the encoder.

2) Model Optimization: In the constructed training dataset, each query corresponds to multiple positive samples and only one negative sample. To explicitly model the relative distribution differences between positive and negative samples, while maximizing the similarity between the query and positive samples and minimizing that of negative sample, we incorporate the contrastive loss function into the model training process, defined as follow:

$$\mathcal{L}_{\text{const}} = -\frac{1}{N}\log \frac{\sum_{i=1}^{N_p} \exp\left(\text{sim}\left(z_q, z_{p_i^+}\right)/\tau\right)}{\sum_{i=1}^{N_p} \exp\left(\text{sim}\left(z_q, z_{p_i^+}\right)/\tau\right) + \exp\left(\text{sim}\left(z_q, z_{p^-}\right)/\tau\right)} \tag{2}$$

where N represents the number of samples in a batch, N_p denotes the number of positive samples related to query q. $z_q, z_{p_i^+}, z_{p^-}$ denote the vector representations encoded by the initial encoder. $\text{sim}(\cdot,\cdot)$ is the cosine similarity function, and τ is the temperature hyper-parameter, used to control the range of similarity. However, due to the sparsity of negative sample, the contrastive loss may suffer from the vanishing gradient problem during training. To alleviate this issue, we additionally introduce the binary cross-entropy (BCE) loss to balance the similarity between positive and negative samples. The BCE loss function is calculated as follows:

$$\mathcal{L}_{\text{BCE}} = -\frac{1}{N_p}\sum_{i=1}^{N_p} \log\left(\text{sim}(z_q, z_{p_i^+})\right) - \log\left(1 - \text{sim}(z_q, z_{p^-})\right) \tag{3}$$

The final training objective is a weighted combination of the contrastive loss and the BCE loss. By jointly optimizing these two objectives, the model progressively sharpens its discrimination between positives and negatives, thereby enhancing capability to encode and represent knowledge. The overall objective is defined as follows:

$$\mathcal{L} = \alpha\mathcal{L}_{\text{const}} + (1-\alpha)\mathcal{L}_{\text{BCE}} \tag{4}$$

where α is a variable hyperparameter whose purpose is to balance their respective contributions to the total loss.

3) Multi-Path Embedding: For queries with multiple candidate paths, retrieval performance is often hindered by noise and redundant information. Although existing embedding models [33], [34] excel at sentence semantic representation, their generalization ability is limited by training on specific corpora, making it difficult to handle unseen sentences or KGs. Hence, their applicability narrows when facing unfamiliar tasks or domains, as confirmed by both theoretical and empirical [11] studies. To address this issue, we pre-train the encoder on a constructed sub-dataset to effectively match candidate knowledge paths to improve LLM outputs. Specifically, each triple (h, r, t) is converted into natural language form. For inverse triples, the

Algorithm 1: Sub-Dataset Construction and Multi-Path Embedding.

Input: KGQA dataset $\mathcal{D}$, question q, knowledge graph $\mathcal{G}$, LLM $\mathcal{M}$, ground-truth answer a
Output: Encoder Encoder_θ

$\mathcal{D}_{\text{sub}} \leftarrow \varnothing$
for $(q, a) \in \mathcal{D}$ **do**
  if $\mathcal{M}(q) = a$ **then**
    continue
  Retrieve candidate paths $Path$ from $\mathcal{G}$
  Generate answers $\hat{a}_{\mathcal{P}} \leftarrow \mathcal{M}(q \,\|\, \text{linearize(Path)})$
  if $\hat{a}_{\mathcal{P}} = a$ **then**
    $\text{set}(p^+) \leftarrow$ paths that yield a
  else
    Construct a single negative p^- using wrong tail
  Build sub-dataset $\mathcal{D}_{\text{sub}}$ using (1)
Optimize loss $\mathcal{L}$ with (2), (3), and (4)
Training multi-path encoder Encoder_θ
return Encoder_θ

path is represented as $Path = (t\ r\ h)$; for example, (France, ˜is capital of, Paris) is expressed as "Paris is capital of France". For forward triples, the representation is directly $Path = (h\ r\ t)$; for instance, (Paris, is capital of, France) is directly expressed as "Paris is capital of France". The model is then trained based on optimization objectives, and the trained encoder is used to quantify the effectiveness of each path. Given question q and its natural language representation $Path$, the multi-path encoder encodes them as:

$$E_q = \text{Encoder}_\theta(q) \tag{5}$$

$$E_{Path} = \text{Encoder}_\theta(Path) \tag{6}$$

where $[q, Path] \in \mathbb{R}^d$, d denotes the dimension of embedding. Algorithm 1 presents the pseudo-code for sub-dataset construction and multi-path embedding.

Since both questions and reasoning paths are encoded using the same encoder, the learned latent semantic similarities can naturally generalize to new queries and candidate paths. This allows the encoder to capture the semantic features of new candidate paths without further training, thereby effectively learning the complex associations between the query and potentially critical paths.

D. Multi-Path Exploration

In existing RAG methods based on cosine similarity retrieval, the reasoning path information often suffers from incomplete coverage or excessive redundancy, which causes the model to overlook some key knowledge when answering questions. To tackle this issue, we introduce multi-path exploration strategy to select the paths within each cluster. This diversified retrieval strategy ensures that the model gathers information from multiple perspectives, avoiding overly simplistic responses and significantly improving the comprehensiveness of the generated answers.

To balance the selection of representative paths from the candidate path set $Path$ while ensuring both path diversity and relevance to the query, we adopt the K-means clustering [35] method. Specifically, the path set is partitioned into c semantic clusters, and the top $\lfloor k/c \rfloor$ reasoning paths from each cluster are selected as the final input prompts for the LLM. Specifically, first divide the path set $Path$ into c clusters:

$$\bigcup_{i=1}^{c} C_i = \text{Clustering}(Path, c) \tag{7}$$

where C_i denotes the i-th cluster, and the cluster size is represented as $m_j = |C_j|$. The total number of selected reasoning paths is k, and the basic quota of each cluster is calculated as:

$$n = \left\lfloor \frac{k}{c} \right\rfloor, \quad r = k \bmod n \tag{8}$$

For each cluster C_i, calculate the semantic similarity score between the query q and each path $Path_i$ within the cluster. The similarity is defined by the cosine similarity of their encoded vector representations, $S = \cos(\mathcal{Z}_q, \mathcal{Z}_{Path})$. Sort the scores in descending order and select the top a_i paths, where a_i is defined as:

$$a_i = \begin{cases} \min\left(m_j, n+1\right), & i \le r \\ \min\left(m_j, n\right), & i > r \end{cases} \tag{9}$$

If the number of paths m_j in a cluster is less than the allocated quota, all paths within that cluster are selected. If the total number of candidate paths $|Path| < k$, all paths are directly output. The final selected reasoning path set is represented as $\mathcal{P}_t$:

$$\mathcal{P}_t = \bigcup_{i=1}^{c} \text{Top}_{a_i}(C_i, S) \tag{10}$$

where $\text{Top}_{a_i}(C_i, S)$ represents the selecting of the top a_i paths with the highest similarity scores from cluster C_i. The detailed steps of multi-path exploration strategy is summarized in Algorithm 2.

By optimizing the parameters c and k, we can flexibly adjust both the selection range of paths and how much knowledge is introduced, enabling a dynamic selection process. This also ensures that the selected paths possess both diversity and relevance, effectively reducing the dominant influence of a single semantic category on the results.

E. Answer Generation

At the answer generation phase, we design a hybrid prompting strategy that integrates both hard and soft prompts to guide the LLM. Specifically, the meticulously designed prompt template is denoted as the hard prompt P_h, while the reasoning path $\mathcal{P}_t$ is termed the soft prompt P_s. To construct the final input instruction, we define a prompt combination function $\text{PM}(\cdot)$ that concatenates or merges the two prompts into a unified instruction prompt I:

$$I = \text{PM}(P_h, P_s) \tag{11}$$

Algorithm 2: Multi-Path Exploration Algorithm.

Input: Question q, number of clusters c, total selections k, candidate paths $Path = \{p_1, \ldots, p_{|Path|}\}$
Output: Reasoning paths $\mathcal{P}_t$
$\mathcal{P}_t \leftarrow \varnothing$
if $|Path| \leq k$ **then**
  return $Path$
Compute the similarity with (5) and (6)
$S \leftarrow \cos(E_q, E_{\text{path}})$
Divide the path set $Path$ into c clusters using (7)
for $i = 1$ **to** c **do**
  $C_i \leftarrow \text{SortDesc}(C_i, S)$
  $m_i \leftarrow |C_i|$
Calculate the basic quota for each cluster with (8)
for $j = 1$ **to** c **do**
  Determine a_j by (9) and select top-a_j paths
  $\mathcal{P}_t \leftarrow \mathcal{P}_t \cup \text{Top}_{a_j}(C_j, S)$
if $|\mathcal{P}_t| < k$ **then**
  $R \leftarrow \text{SortDesc}(Path \setminus \mathcal{P}_t, S)$
  $\mathcal{P}_t \leftarrow \mathcal{P}_t \cup$ first $(k - |\mathcal{P}_t|)$ items of R
return $\mathcal{P}_t$

Given a natural language question q, we insert both P_h and P_s into their designated positions within q, thereby forming the augmented instruction $[q, I]$ that serves as the input to the LLM. Let the answer sequence be $Y = (y_1, y_2, \ldots, y_r)$, where r is the sequence length. The conditional probability of generating Y is modeled in an auto-regressive manner:

$$P(Y \mid q, I) = \prod_{i=1}^{r} P(y_i \mid y_{<i}, q, I; \theta_{LLM}) \tag{12}$$

where $y_{<i} = (y_1, \ldots, y_{i-1})$ represents the prefix subsequence, and θ_{LLM} denotes the fixed parameters of the pre-trained LLM.

This formulation allows us to leverage both the task-specific constraints encoded in P_h and the structured reasoning knowledge embedded in P_s, without updating the parameters of the LLM. By explicitly injecting reasoning paths into the prompt, our approach effectively mitigates hallucination and enhances the reliability of LLMs in domain-specific reasoning tasks.

V. Experiments

A. Experimental Setup

1) Datasets: To assess the effectiveness of ELMK on multi-hop knowledge-intensive reasoning, we conducted comparative experiments on four representative KGQA benchmarks: MetaQA [36], FACTKG [37], WebQSP [38] and CWQ [39]. MetaQA is a widely used multi-hop QA dataset with three subsets, namely 1-hop, 2-hop and 3-hop, and contains over 400,000 question answer pairs. FACTKG is a KG fact verification dataset with about 108 K claims, covering five reasoning types: one-hop, conjunction, existence, multi-hop and negation. WebQSP mainly consists of relatively simple 1-hop and 2-hop questions with 4737 instances, while CWQ contains about 34,689 more challenging questions that typically require up to 4-hop reasoning over the knowledge graph.

2) Baselines: In our exploration of the MetaQA dataset, we select four categories of baseline methods for comparison: Embedding-based methods, retrieval-based methods, vanilla LLMs, and KG-enhanced LLM methods. Specifically, the embedding-based baselines include KV-Mem [40], Embed-KGQA [41], and NSM [42], all of which operate in a fully supervised fashion. The retrieval-based baselines include Graft-Net [43] and UniKGQA [44], which operate in a fully supervised fashion. The vanilla LLMs include GPT-3.5-turbo [5], GPT-4o-mini [45], LLaMA2-7B [46], and DeepSeek-v3 [47], which implement a 4-shot configuration. The KG-enhanced LLM methods encompass KG-GPT [18] and KELP [17], are used to evaluate the model performance across employing 4-shot, 8-shot, 12-shot configurations. For the evaluation of FACTKG, we use the same baseline as KG-GPT. To enable a more comprehensive evaluation on WebQSP and CWQ, we additionally include ReaRev [48], KD-CoT [49], StructGPT [50], ToG [13], PoG [14] and RoG [31] as baselines. To ensure fairness, the selected KG-enhanced LLM methods do not involve fine-tuning the LLM.

3) Evaluation Metrics: Consistent with earlier works [5], [18], we adopt accuracy and F1 score as the primary metrics. Accuracy deems a prediction correct when it matches at least one ground-truth answer. Given that a question may have multiple correct answers, we further introduce F1 to measure the overlap between the predicted and gold answers, effectively balancing the precision and recall of the predictions.

4) Implementation Details: In our experimental setup, we adopt different configurations of LLMs and knowledge encoders for different datasets. For MetaQA and FACTKG, we use GPT-4o-mini as the LLM backbone and employ the pre-trained all-mpnet-v2 [34] model as the knowledge encoder. In contrast, for the more challenging reasoning tasks in WebQSP and CWQ, we use GPT-3.5-turbo as the LLM backbone and fully fine-tune LLaMA2-7B to serve as the knowledge encoder. Training is conducted using the AdamW optimizer with the learning rate of 5e-5, a loss function weight of $\alpha = 0.85$, the temperature parameter $\tau = 0.1$, the batch size is set to 64, and the total of 60 training epochs. All experiments are conducted on 4 NVIDIA A100 40G GPUs. In the multi-path exploration phase, for MetaQA 1-hop, 2-hop, 3-hop, we set $k \in (20, 40, 80)$ and $c \in (3, 2, 4)$, respectively. For FACTKG, we set $k = 15$ and $c = 4$ to generate reasoning paths for each query, while for WebQSP and CWQ we set k to 8 and 6, and c to 5 and 4, respectively.

B. Main Results Comparison

We compare ELMK against other baselines on KGQA benchmarks, with a particular focus on the few-shot learning settings, as shown in Tables III, and IV. The experimental results demonstrate that ELMK not only performs strongly on simple single-hop reasoning problems but also exhibits superior performance in multi-hop complex queries. Specifically, compared with state-of-the-art method, ELMK achieves best accuracy

TABLE III
PERFORMANCE COMPARISON OF DIFFERENT BASELINES ON METAQA DATASET. THE BEST RESULTS ARE MARKED IN BOLD, AND * MARKS RESULTS OBTAINED BY REPRODUCING THE METHODS.

Types	Methods	Training Strategy	MetaQA 1-hop		MetaQA 2-hop		MetaQA 3-hop	
			Accuracy	F1	Accuracy	F1	Accuracy	F1
Embedding	KV-Mem	Supervised learning	96.2	–	82.7	–	48.9	–
	EmbedKGQA		97.5	–	98.8	–	94.8	–
	NSM		97.1	–	99.9	–	98.9	–
Retrieval	UniKGQA	Supervised learning	97.5	–	99.0	–	99.1	–
	GraftNet		97.0	–	94.8	–	77.7	–
LLM	LLaMA2-7B	4-shot	50.7	46.1	43.4	21.7	22.8	16.4
	GPT-3.5-turbo		61.6	56.5	56.1	41.7	37.9	18.3
	GPT-4o-mini		61.2	57.1	56.6	42.1	38.8	17.5
	DeepSeek-v3		62.7	58.7	55.5	41.6	40.1	20.6
LLM+KG	KG-GPT*	4-shot	95.4	70.5	94.1	52.5	73.1	25.3
		8-shot	96.1	71.1	94.8	53.9	75.2	26.5
		12-shot	96.3	71.2	95.2	53.6	**90.4**	28.9
	KELP*	4-shot	96.7	70.7	85.9	55.7	66.7	25.0
		8-shot	96.4	78.6	87.6	57.1	75.6	25.6
		12-shot	89.2	65.4	88.4	57.6	75.0	26.3
	ELMK	4-shot	**99.9**	94.6	97.9	86.5	78.7	**32.6**
		8-shot	98.7	93.7	**98.2**	**86.6**	80.3	31.5
		12-shot	98.7	**94.9**	98.1	86.5	79.2	30.8

TABLE IV
PERFORMANCE COMPARISON OF FACTKG DATASET

Input Type	Methods	Training Strategy	FACTKG	
			Accuracy	F1
Claim Only	BERT	Supervised learning	65.2	–
	BlueBERT		59.9	–
	Flan-T5	zero-shot	54.2	–
	GPT-4o-mini	4-shot	58.3	46.7
With Evidence	KG-GPT*	4-shot	58.6	56.3
		8-shot	66.5	61.8
		12-shot	70.1	64.1
	KELP*	4-shot	67.4	62.5
		8-shot	67.9	64.7
		12-shot	68.6	63.8
	ELMK	4-shot	74.4	70.1
		8-shot	74.7	70.3
		12-shot	**76.2**	**71.4**

TABLE V
PERFORMANCE COMPARISON OF WEBQSP AND CWQ

Types	Methods	WebQSP		CWQ	
		Accuracy	F1	Accuracy	F1
Graph Reasoning	GraftNet	66.4	60.4	36.8	32.7
	NSM	68.7	62.8	47.6	42.4
	UniKGQA	77.2	72.2	51.2	49.1
	ReaRev	76.4	70.9	52.9	47.8
LLM	LLaMA2-7B	56.4	36.5	28.4	21.4
	GPT-3.5-turbo	63.1	39.7	39.2	32.8
	GPT-4o-mini	63.8	40.5	39.6	32.5
	DeepSeek-v3	64.0	43.9	41.1	33.8
KG+LLM	KD-CoT	68.6	52.5	55.7	–
	StructGPT	72.6	63.7	54.3	49.6
	ToG	76.2	–	57.1	–
	PoG*	80.2	62.4	58.5	49.9
	RoG	**85.7**	70.8	62.6	56.2
	ELMK	83.6	**72.5**	**63.0**	**60.6**

surpassing KELP by 3.2%, 10.6%, 4.7%, and F1 improvement of up to 29.5%, 29.5%, and 7.6% across the three MetaQA subtasks. On the more challenging FACTKG, ELMK improves both accuracy and F1 by 7.6% over the SOTA baseline. However, the gains are slightly smaller than that on MetaQA, because MetaQA provides clear topic entities and enumerable multi-hop paths, which better match ELMK's multi-path exploration strategy. In contrast, FACTKG is a fact verification task where reasoning chains are often implicit and may involve complex logic such as negation and existential statements, making path alignment more difficult and thus partially attenuating ELMK's advantages. Furthermore, as shown in Table V, ELMK achieves optimal results on both WebQSP and CWQ, with particularly notable advantages in terms of the F1. Compared with KG+LLM methods, ELMK attains the highest F1 score of 72.5 on WebQSP and 60.6 on CWQ. This indicates that the multi-path exploration strategy adopted by ELMK enhances the diversity of reasoning paths, thereby enabling LLMs to generate more complete and comprehensive ground-truth answers.

Notably, LLM-only approaches suffer substantial performance degradation due to hallucinations and a lack of prior knowledge. In contrast, KG-GPT, KELP and PoG improve answer accuracy by leveraging KG to enhance LLM reasoning. However, F1 score is significantly lower than ELMK on questions with multiple correct answers. Because the method like

TABLE VI
COST STATISTICS OF ELMK AND TWO BASELINES

Methods	Evaluation	MetaQA 1-hop	MetaQA 2-hop	MetaQA 3-hop	FACTKG
KG-GPT	LLM Calls	3	3	3	3
	Avg. Time (s)	5.20	5.65	18.66	6.28
	Avg. Tokens	1786	2473	5661	2508
KELP	LLM Calls	4	4	4	4
	Avg. Time (s)	1.59	2.50	17.56	4.85
	Avg. Tokens	2325	2800	8587	3204
ELMK	LLM Calls	2	2	2	2
	Avg. Time (s)	1.22	2.13	13.40	4.21
	Avg. Tokens	1222	1970	5646	1399

KELP typically considers only a single candidate path during encoding, whereas ELMK uses a pre-train multi-path encoder to simultaneously encode multiple candidate paths and selects semantically diverse reasoning paths based on clustering results, thereby increasing knowledge diversity and improving the LLM understanding of KG information. Furthermore, in few-shot scenarios, ELMK surpasses most fully supervised models and achieves highest accuracy on these datasets. In particular, under the 4-shot setting, ELMK improved the accuracy of tasks on the MetaQA by 3.7%, 15.2%, and 29.8%, respectively compared to KV-Mem, demonstrating that our method maintains high performance even with only a limited number of samples.

C. Details Analysis

1) Efficiency Analysis: To validate the efficiency of our method, Table VI presents the detailed comparison of computational overhead between ELMK and two representative baselines (i.e., KG-GPT and KELP). Specifically, we measure the average inference time per request, the average number of LLM calls, and the average number of tokens required per request. The results show that ELMK achieves the lowest runtime and token consumption, while limiting the number of LLM calls to only two per request, resulting in a significantly lower execution cost compared with KG-GPT and KELP. Since KG-GPT adopts a step-by-step reasoning paradigm that requires three LLM calls, as the complexity of the questions increases, each call must incorporate a large amount of contextual knowledge as input, leading to sharp growth in token usage and runtime. In contrast, KELP requires four LLM calls in total for dataset construction and model inference, which dramatically increases token consumption per request and yields higher inference time than ELMK. Combining the results from Tables III and IV, we conclude that ELMK significantly improves efficiency without sacrificing reasoning performance. Furthermore, ELMK selects multiple reasoning paths from different clusters through clustering, without requiring additional LLM calls or token inputs, further demonstrating its efficiency in integrating LLMs.

2) Parameter Sensitivity Analysis: We analyze the impact of different k on the performance of ELMK during the multi-path exploration phase and conduct experiments with various k values. The results are shown in Fig. 4. F1 and recall of ELMK increase with the rise in k, as a higher k allows the retrieval

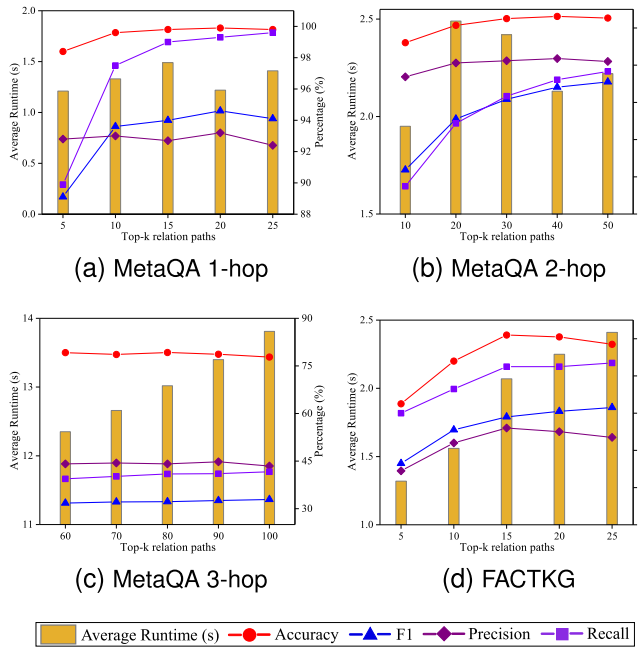

Fig. 4. Performance of ELMK on different k values.

of more reasoning paths, thereby covering a greater number of correct answers. However, increasing k also introduces a large amount of irrelevant or noisy context, which not only decreases accuracy and precision but also increases retrieval time and LLM token consumption, rendering the reasoning results unreliable. This highlights the importance of using the multi-path exploration strategy to select the correct reasoning paths. Taking the FACTKG dataset as an example, performance gradually improves as k increases, reaching a peak around $k = 15$. Beyond this point, as k continues to increase, accuracy and precision begin to decline while the average runtime continues to increase. A comparable trend holds on MetaQA, indicating that selecting an appropriate k is crucial to balance accuracy and efficiency. Therefore, in the experiments on the three subtasks of MetaQA and FACTKG, we set k to 20, 40, 80, and 15, respectively, enabling ELMK to achieve a well-balanced trade-off between reasoning performance and efficiency.

We also analyze the impact of different cluster numbers c on the performance of ELMK. As shown in Fig. 5, when $c = 5$, ELMK exhibits performance degradation across all datasets. To achieve the best reasoning performance, we set c to 3, 2, 4, and 4 on the MetaQA and FACTKG datasets, respectively.

3) Reasoning Faithfulness Analysis: To assess the effectiveness of different methods in terms of answer coverage, we compare the overlap between the generated correct answers and the ground-truth answers. Specifically, the overlap ratio [13] is defined as the proportion of the number of answers in the intersection of the two sets to the total number of ground-truth answers. Fig. 6 presents the overlap ratio distributions of ELMK and baselines across four datasets. On MetaQA 1-hop, ELMK achieves the highest complete overlap rate of 98.0%, compared with 64.1% for KG-GPT and 57.6% for KELP, with the rest distributed across lower intervals. This indicates that ELMK

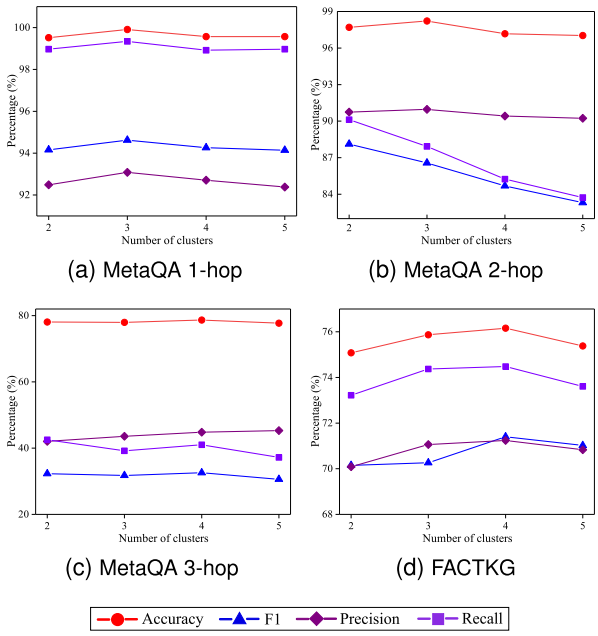

Fig. 5. Performance of ELMK with different cluster numbers c.

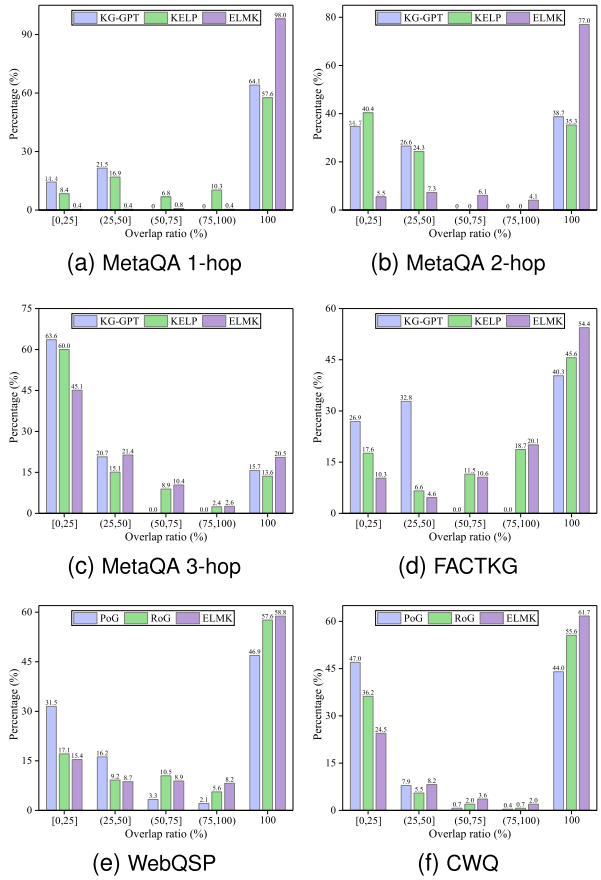

Fig. 6. The answers overlap ratio of ELMK.

can generate answers highly consistent with the ground truth in 1-hop reasoning tasks. Although the distribution becomes more dispersed on MetaQA 2-hop and FactKG, ELMK still attains complete overlap for more than half of the samples. In contrast, on MetaQA 3-hop, ELMK achieves a complete overlap ratio of 20.5%, significantly higher than KG-GPT and KELP's 15.7% and 13.6%, while exhibiting a smaller proportion in lower overlap intervals. In addition, the proportion of KG-GPT in the two intervals of (50,75] and (75,100) is 0, implying it tends to pick a single fully matching path and has difficulty covering multiple correct candidate paths. Similarly, KELP pruning strategy retains only the single reasoning path with the highest relevance. Comparable advantages are observed on WebQSP and CWQ, where ELMK outperforms PoG and RoG in complete overlap, reaching 58.8% on WebQSP and 61.7% on CWQ, while reducing low overlap to 15.4% on WebQSP, compared with 31.5% for PoG and 17.1% for RoG. In summary, ELMK outperforms the baselines across all datasets. Its advantage stems from the multi-path encoding and multi-path exploration strategy, which clusters candidate paths and selects multiple representative paths from each cluster to guide LLM reasoning, thereby improving both the diversity and accuracy of reasoning.

D. Ablation Studies

We run ablations to assess the contribution of each component, comparing three variants: (1) *w/o Training*, where we remove the pre-training multi-path encoder and use an off-the-shelf general-purpose encoder; (2) *w/o Clustering*, where we remove the multi-path exploration strategy and randomly select reasoning paths from the reasoning subgraph; (3) *w/ Direct retrieve*, directly select the top-k paths from the highest scores to form the path set. The experimental results are shown in Table VII.

The results demonstrate that the complete ELMK achieves the best performance on all datasets, and removing any component leads to performance degradation, which strongly validates the effectiveness of our design. Specifically, although removing the pre-trained encoder causes a decline in performance, the drop is relatively small. The findings underscore the necessity of training the encoder to deepen knowledge comprehension and representation, enabling it to more accurately capture the complex structured information within the KG. In contrast, eliminating the multi-path exploration strategy and replacing it with random retrieval causes a significant performance loss, particularly with a notable drop in the F1 score. This is because clustering allows the selection of the most relevant reasoning path from each cluster, thereby improving the comprehensiveness and accuracy of the answers, especially for questions with multiple correct answers, further highlighting the importance of the multi-path exploration strategy. Moreover, relying solely on semantic-based direct retrieval fails to provide satisfactory responses for multi-hop reasoning tasks, further confirming the necessity of our proposed multi-path exploration strategy.

E. Performance With Different LLMs

In this subsection, we evaluate the performance of integrating ELMK with different LLMs to enhance reasoning performance on the MetaQA dataset. The four LLMs used are LLaMA2-7B, GPT-3.5-turbo, GPT-4o-mini, and DeepSeek-v3, with

TABLE VII
ABLATION STUDIES OF ELMK FRAMEWORK

Methods	MetaQA 1-hop		MetaQA 2-hop		MetaQA 3-hop		FACTKG		WebQSP		CWQ	
	Accuracy	F1	Accuracy	F1	Accuracy	F1	Accuracy	F1	Accuracy	F1	Accuracy	F1
ELMK	**99.87**	**94.61**	**97.87**	**86.50**	**78.67**	**32.57**	**76.16**	**71.40**	**83.62**	**72.48**	**63.04**	**60.63**
ELMK w/o Training	90.72	86.07	89.33	66.84	56.56	21.34	72.35	65.19	67.51	55.68	45.33	42.25
ELMK w/o Clustering	72.55	58.06	70.23	43.66	48.58	15.25	50.36	42.74	65.80	50.63	41.69	36.41
ELMK w/ Direct retrieve	82.43	64.74	80.37	58.92	51.42	18.86	57.80	46.71	69.07	51.98	42.22	39.03

TABLE VIII
THE PERFORMANCE COMPARISON OF INTEGRATING DIFFERENT LLMS INTO THE ELMK METHOD FOR REASONING

method	MetaQA 1-hop				MetaQA 2-hop				MetaQA 3-hop			
	Accuracy	F1	Precision	Recall	Accuracy	F1	Precision	Recall	Accuracy	F1	Precision	Recall
LLaMA2-7B	50.73	46.14	44.88	48.36	43.41	21.74	20.32	23.46	22.82	16.44	18.66	15.57
LLaMA2-7B+ELMK	**89.82**	**88.31**	**88.07**	**89.22**	**86.75**	**80.34**	**80.68**	**80.09**	**70.02**	**28.71**	**36.45**	**35.22**
GPT-3.5-turbo	61.55	56.54	56.23	58.34	56.10	41.68	41.07	43.24	37.91	18.33	30.10	24.04
GPT-3.5-turbo+ELMK	**99.60**	**96.14**	**95.13**	**99.13**	**97.48**	**87.32**	**91.34**	**87.87**	**78.77**	**32.06**	**43.91**	**41.19**
GPT-4o-mini	61.24	57.11	56.68	58.76	56.55	42.09	40.17	44.08	38.82	17.51	30.08	23.33
GPT-4o-mini+ELMK	**99.87**	**94.61**	**93.15**	**99.30**	**97.90**	**86.53**	**91.06**	**87.65**	**78.67**	**32.57**	**44.79**	**41.03**
DeepSeek-v3	62.67	58.68	57.61	59.64	55.47	41.64	40.22	42.88	40.12	20.56	28.37	24.44
DeepSeek-v3+ELMK	**99.70**	**96.20**	**95.15**	**99.25**	**98.80**	**89.02**	**95.83**	**87.90**	**81.73**	**43.56**	**64.99**	**42.21**

evaluation metrics including accuracy, F1, precision, and recall. The detailed experimental results are presented in Table VIII. The results show that incorporating the multi-path optimization strategy of ELMK leads to notable improvements for all LLMs. Specifically, taking the MetaQA 1-hop dataset as an example, the accuracy of the four LLMs increases by 39.09%, 38.05%, 38.63%, and 37.03%, respectively, while F1 scores improve by 42.17%, 39.60%, 37.50%, and 37.52%. These findings clearly demonstrate that ELMK effectively enhances LLM performance on KGQA tasks and can be seamlessly integrated with different LLMs without requiring additional training. Therefore, ELMK can be regarded as a universally effective enhancement strategy that directly alleviates hallucination problem in the LLM reasoning process.

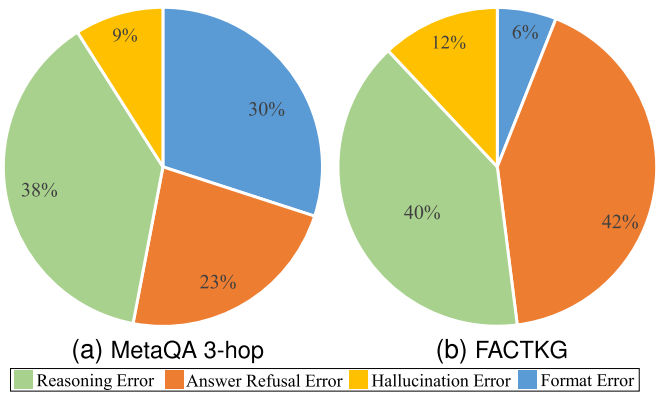

Fig. 7. Proportion of different error categories of ELMK in the MetaQA 3-hop and FACTKG datasets.

F. Error Analysis

To examine how LLMs interact with KG, we perform an error analysis on the MetaQA 3-hop and FACTKG datasets, taking 100 random error cases per dataset. Then, we categorize errors into four types: (1) Reasoning error, (2) answer refusal error, (3) hallucination error, and (4) format error. We present the statistical results in Fig. 7. For MetaQA 3-hop, the proportions of reasoning error, answer format error, and answer refusal error are relatively close, with format error primarily arising from the LLM incorporating special symbols during answer generation, causing misalignment with the gold answers. For FACTKG, the main error types are reasoning error and answer refusal error, each accounting for 40%, which is noticeably higher than on MetaQA. This indicates that in the FACTKG, providing KG paths can improve performance to some extent, but when evidence is insufficient, LLMs still tend to refuse to answer or make reasoning mistakes, thereby limiting the overall performance gains.

G. Case Study

We also provide case studies to demonstrate the advantages of ELMK in terms of reasoning interpretability and answer comprehensiveness. As shown in Table IX, the first case demonstrates that KELP produces an incorrect answer due to hallucinated reasoning paths. In contrast, ELMK is able to retrieve the correct reasoning paths from KG for reasoning, while providing clear and interpretable answer generation process. In the second case, although KELP can generate relevant answers, it fails to fully cover all correct answers in complex tasks. Conversely, ELMK performs faithful reasoning through comprehensive and accurate reasoning paths. Hence, ELMK not only effectively eliminates hallucinations and generate comprehensive answers, but also leverage complex information from KG to achieve faithful and interpretable reasoning. In addition, to provide a more comprehensive view of the limitations and robustness of ELMK. Case 3 presents a failure example, when candidate

TABLE IX
EXAMPLES OF ANSWERS GENERATED BY ELMK AND KELP ON THE METAQA, INCLUDING A FAILURE CASE OF ELMK. RED DENOTES THE INCORRECT REASONING PATHS AND ANSWER, WHILE BOLD INDICATES THE CORRECT ANSWERS.

Case 1: Incorrect answer of KELP and correct as well as interpretable answer of ELMK.	
Question	what genres do the movies that share screenwriters with Afterwards fall under?
Topic entity	Afterwards
Ground-truth	{Drama}
KELP	# Reasoning Path: {Afterwards} → written → {Gilles Bourdos} → ˜directed by → {Udaan} → has genre → {Comedy} # Answer: {Comedy}
ELMK	# Reasoning Path: {Afterwards} → written → {Gilles Bourdos} → ˜written by → {Renoir} → has genre → {Drama} # Answer: {**Drama**}
Case 2: Incomplete answer of KELP and comprehensive as well as faithful reasoning of ELMK.	
Question	what were the release dates of Chico Marx starred movies?
Topic entity	Chico Marx
Ground-truth	{1931, 1949, 1946, 1935, 1952}
KELP	# Reasoning Path: {Chico Mar} → starred actors → {A Night in Casablanca} → release year → {1946} # Answer: {**1946**}
ELMK	# Reasoning Paths: {Chico Mar} → starred actors → Love Happy → release year → {1949} {Chico Mar} → starred actors → {Monkey Business} → release year → {1931} {Chico Mar} → starred actors → {Monkey Business} → release year → {1952} {Chico Mar} → starred actors → {A Night at the Opera} → release year → {1935} {Chico Mar} → starred actors → {A Night in Casablanca} → release year → {1946} # Answer: {**1931, 1949, 1946, 1935, 1952**}
Case 3: The failure of ELMK reasoning stems from the high semantic similarity of relations along the reasoning paths.	
Question	who co-wrote films with Sam Shepard?
Topic entity	Sam Shepard
Ground-truth	{L.M. Kit Carson, Wim Wenders}
Correct paths	# Reasoning Path: {Sam Shepard} → written → {Paris, Texas} → ˜written by → {L.M. Kit Carson} {Sam Shepard} → written → {Don't Come Knocking} → ˜written by → {Wim Wenders}
ELMK	# Reasoning Paths: {Sam Shepard} → starred actors → Blind Horizon → ˜written by → {Steve Tomlin} {Sam Shepard} → starred actors → The Right Stuff → ˜written by → {Tom Wolfe} # Answer: {Steve Tomlin, Tom Wolfe}

relations are highly similar in semantics, ELMK is prone to interference, leading to the selection of incorrect reasoning paths and erroneous answers. This result indicates that path selection under high semantic similarity remains a typical challenge for LLMs.

VI. CONCLUSION

In this paper, we propose ELMK, a novel LLM reasoning method that effectively integrates the explicit factual information from KG with the implicit knowledge embedded in LLM, enabling comprehensive and interpretable reasoning. The ELMK method employs a three-stage multi-path optimization framework of retrieval–embedding–reasoning. First, DFS is used to efficiently retrieve the reasoning graph from the KG. Then, a multi-path encoder is trained to semantically quantify candidate paths with potential influence. Finally, a multi-path exploration strategy is applied to further optimize the selection of knowledge paths. ELMK not only enhances the reasoning ability of LLMs by retrieving structured knowledge from KG but also offers plug-and-play compatibility with any LLM. Experiments on four widely available KGQA datasets demonstrate that ELMK markedly surpass existing baselines, effectively improving both answer accuracy and coverage. Due to the computational resources required for constructing datasets and pre-training encoders, in future work, we plan to explore how to adaptively meet encoding needs across various scenarios without the need to train the encoder.

REFERENCES

[1] A. Algherairy and M. Ahmed, "A review of dialogue systems: Current trends and future directions," *Neural Comput. Appl.*, vol. 36, no. 12, pp. 6325–6351, 2024.
[2] G. Xiong, J. Bao, and W. Zhao, "Interactive-KBQA: Multi-turn interactions for knowledge base question answering with large language models," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2024, pp. 10561–10582.
[3] H. Luo et al., "ChatKBQA: A generate-then-retrieve framework for knowledge base question answering with fine-tuned large language models," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2024, pp. 2039–2056.
[4] W. Cheng, Y. Wu, and W. Hu, "Dataflow-guided retrieval augmentation for repository-level code completion," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2024, pp. 7957–7977.
[5] Y. Wen, Z. Wang, and J. Sun, "MindMap: Knowledge graph prompting sparks graph of thoughts in large language models," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2024, pp. 10370–10388.
[6] T. Ao et al., "LightPROF: A lightweight reasoning framework for large language model on knowledge graph," in *Proc. AAAI Conf. Artif. Intell.*, 2025, pp. 23424–23432.

[7] D. Yang et al., "IM-RAG: Multi-round retrieval-augmented generation through learning inner monologues," in *Proc. 47th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval.*, 2024, pp. 730–740.
[8] P. Lewis et al., "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 9459–9474.
[9] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, "Self-RAG: Learning to retrieve, generate, and critique through self-reflection," in *Proc. 12th Int. Conf. Learn. Representations*, 2024.
[10] H. Huang et al., "Retrieval-augmented generation with hierarchical knowledge," 2025, *arXiv:2503.10150*.
[11] Y. Yu et al., "RankRAG: Unifying context ranking with retrieval-augmented generation in LLMs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2024, pp. 121156–121184.
[12] S. Pan, L. Luo, Y. Wang, C. Chen, J. Wang, and X. Wu, "Unifying large language models and knowledge graphs: A roadmap," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 7, pp. 3580–3599, Jul. 2024.
[13] J. Sun et al., "Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph," in *Proc. 12th Int. Conf. Learn. Representations*, 2024.
[14] L. Chen, P. Tong, Z. Jin, Y. Sun, J. Ye, and H. Xiong, "Plan-on-graph: Self-correcting adaptive planning of large language model on knowledge graphs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2024, pp. 37665–37691.
[15] X. Ye, S. Yavuz, K. Hashimoto, Y. Zhou, and C. Xiong, "RNG-KBQA: Generation augmented iterative ranking for knowledge base question answering," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2022, pp. 6032–6043.
[16] I. Abdelaziz, S. Ravishankar, P. Kapanipathi, S. Roukos, and A. Gray, "A semantic parsing and reasoning-based approach to knowledge base question answering," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 15985–15987.
[17] H. Liu, S. Wang, Y. Zhu, Y. Dong, and J. Li, "Knowledge graph-enhanced large language models via path selection," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2024, pp. 6311–6321.
[18] J. Kim, Y. Kwon, Y. Jo, and E. Choi, "KG-GPT: A general framework for reasoning on knowledge graphs using large language models," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2023, pp. 9410–9421.
[19] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, "Large language models are zero-shot reasoners," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2022, pp. 22199–22213.
[20] Y. Fu, H. Peng, A. Sabharwal, P. Clark, and T. Khot, "Complexity-based prompting for multi-step reasoning," in *Proc. 11th Int. Conf. Learn. Representations*, 2023.
[21] J. Wei et al., "Chain-of-thought prompting elicits reasoning in large language models," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2022, pp. 24824–24837.
[22] Y. Zhang, X. Wang, L. Wu, and J. Wang, "Enhancing chain of thought prompting in large language models via reasoning patterns," in *Proc. AAAI Conf. Artif. Intell.*, 2025, pp. 25985–25993.
[23] L. Luo, Z. Zhao, G. Haffari, D. Phung, C. Gong, and S. Pan, "GFM-RAG: Graph foundation model for retrieval augmented generation," 2025, *arXiv:2502.01113*.
[24] M. Galkin, X. Yuan, H. Mostafa, J. Tang, and Z. Zhu, "Towards foundation models for knowledge graph reasoning," in *Proc. 12th Int. Conf. Learn. Representations*, 2024.
[25] H. Wang, F. Zhang, M. Zhao, W. Li, X. Xie, and M. Guo, "Multi-task feature learning for knowledge graph enhanced recommendation," in *Proc. World Wide Web Conf.*, 2019, pp. 2000–2010.
[26] M. Gao, J. Li, C. Chen, Y. Li, J. Zhang, and Z. Zhan, "Enhanced multi-task learning and knowledge graph-based recommender system," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 10, pp. 10281–10294, Oct. 2023.
[27] X. Wang et al., "KEPLER: A unified model for knowledge embedding and pre-trained language representation," *Trans. Assoc. Comput. Linguistics.*, vol. 9, pp. 176–194, 2021.
[28] T. Sun et al., "CoLAKE: Contextualized language and knowledge embedding," in *Proc. 28th Int. Conf. Comput. Linguistics*, 2020, pp. 3660–3670.
[29] W. Liu et al., "K-BERT: Enabling language representation with knowledge graph," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 2901–2908.
[30] T. Zhang et al., "DKPLM: Decomposable knowledge-enhanced pre-trained language model for natural language understanding," in *Proc. AAAI Conf. Artif. Intell.*, 2022, pp. 11703–11711.
[31] L. Luo, Y.-F. Li, G. Haffari, and S. Pan, "Reasoning on graphs: Faithful and interpretable large language model reasoning," in *Proc. Int. Conf. Learn. Representations*, 2024.
[32] R. Liu et al., "Ontology-guided reverse thinking makes large language models stronger on knowledge graph question answering," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2025, pp. 15269–15284.
[33] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using siamese BERT-networks," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2019, pp. 3980–3990.
[34] SBERT, "Sentence-transformers all-mpnet-base-v2," 2021. [Online]. Available: https://huggingface.co/sentence-transformers/all-mpnet-base-v2
[35] J. Liao, X. Wu, Y. Wu, and J. Shu, "K-NNDP: K-means algorithm based on nearest neighbor density peak optimization and outlier removal," *Knowl. Based Syst.*, vol. 294, 2024, Art. no. 111742.
[36] Y. Zhang, H. Dai, Z. Kozareva, A. Smola, and L. Song, "Variational reasoning for question answering with knowledge graph," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 6069–6076.
[37] J. Kim, S. Park, Y. Kwon, Y. Jo, J. Thorne, and E. Choi, "FactKG: Fact verification via reasoning on knowledge graphs," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2023, pp. 16190–16206.
[38] A. Talmor and J. Berant, "The web as a knowledge-base for answering complex questions," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2018, pp. 641–651.
[39] W.-t. Yih, M. Richardson, C. Meek, M.-W. Chang, and J. Suh, "The value of semantic parse labeling for knowledge base question answering," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 201–206.
[40] K. Xu, Y. Lai, Y. Feng, and Z. Wang, "Enhancing key-value memory neural networks for knowledge based question answering," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2019, pp. 2937–2947.
[41] A. Saxena, A. Tripathi, and P. P. Talukdar, "Improving multi-hop question answering over knowledge graphs using knowledge base embeddings," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 4498–4507.
[42] G. He, Y. Lan, J. Jiang, W. X. Zhao, and J. Wen, "Improving multi-hop knowledge base question answering by learning intermediate supervision signals," in *Proc. 14th ACM Int. Conf. Web Search Data Mining*, 2021, pp. 553–561.
[43] H. Sun, B. Dhingra, M. Zaheer, K. Mazaitis, R. Salakhutdinov, and W. W. Cohen, "Open domain question answering using early fusion of knowledge bases and text," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 4231–4242.
[44] J. Jiang, K. Zhou, X. Zhao, and J. Wen, "UniKGQA: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graph," in *Proc. 11th Int. Conf. Learn. Representations*, 2023.
[45] OpenAI, "GPT-4o," 2024. [Online]. Available: http://openai.com/index/hello-gpt-4o/
[46] H. Touvron, L. Martin, R. Stojnic, S. Edunov, and T. Scialom, "Llama 2: Open foundation and fine-tuned chat models," 2023, *arXiv:2307.09288*.
[47] D. Guo, D. Yang, H. Zhang, J. Song, W. Liang, and Z. Zhang, "DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning," *Nature*, vol. 645, pp. 633–638, 2025.
[48] C. Mavromatis and G. Karypis, "ReaRev: Adaptive reasoning for question answering over knowledge graphs," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2022, pp. 2447–2458.
[49] K. Wang et al., "Knowledge-driven CoT: Exploring faithful reasoning in LLMs for knowledge-intensive question answering," 2023, *arXiv:2308.13259*.
[50] J. Jiang, K. Zhou, Z. Dong, K. Ye, X. Zhao, and J. Wen, "StructGPT: A general framework for large language model to reason over structured data," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2023, pp. 9237–9251.

Jiyong Liao received the MS degree from the Faculty of Information Engineering and Automation, Kunming University of Science and Technology, China, in 2021. He is currently working toward the PhD degree in computer science and technology with Hunan University, China. His research interests include large language model and artificial intelligence.

Chubo Liu (Member, IEEE) received the BS and PhD degrees in computer science and technology from Hunan University, China, in 2011 and 2016, respectively. He is currently a full professor of computer science and technology with Hunan University. His research interests include parallel and distributed computing, computer architecture, artificial intelligence, and approximation and randomized algorithms. He has published more than 40 papers in journals and conferences such as *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Cloud Computing*, *IEEE Transactions on Mobile Computing*, ISCA, DAC.

Yan Ding (Member, IEEE) received the PhD degree in computer science from Hunan University, China, in 2021. He is currently an assistant professor with Hunan University. His research interests include parallel computing, mobile edge computing, Big Data, artificial intelligence, and architecture. He has published eight papers in journals and conferences, including Design Automation Conference, *IEEE Transactions on Parallel and Distributed Systems*, *IEEE Transactions on Services Computing*, *IEEE Transactions on Industrial Informatics*, *Journal of Parallel and Distributed Computing*, *Computers & Security*, and the 17th IEEE International Symposium on Parallel and Distributed Processing with Applications (IEEE ISPA 2019).

Haotian Wang received the BS degree from the School of Information Engineering, Nanchang University, China, in 2018, and the PhD degree in computer science from Hunan University, China, in 2023. He is currently working as a postdoctoral fellow with Hunan University, China. He has published more than 10 papers in journals and conferences such as *IEEE Transactions on Parallel and Distributed Systems*. He previously completed a one-year joint PhD program with Nanyang Technological University, and he is an ACM member. His research interests include parallel computing and artificial intelligence.

Zhuo Tang (Member, IEEE) received the PhD degree in computer science from the Huazhong University of Science and Technology, China, in 2008. Currently, he is a professor with the College of Computer Science and Electronic Engineering, Hunan University. He is also the chief engineer with the National Supercomputing Center in Changsha. He has published almost 120 journal articles and book chapters. His research interests include distributed computing system, parallel processing for Big Data, including distributed machine learning, and resources scheduling and management in these areas.

Kenli Li (Senior Member, IEEE) received the MS degree in mathematics from Central South University, China, in 2000, and the PhD degree in computer science from the Huazhong University of Science and Technology, China, in 2003. He was a visiting scholar with the University of Illinois at Urbana-Champaign from 2004 to 2005. He is a full professor of computer science and technology with Hunan University. His research interests include parallel and distributed processing, high-performance computing for Big Data and artificial intelligence, etc. He has published more than 300 papers in international conferences and journals. He is currently served on the editorial boards of *IEEE Transactions on Computers*.

Keqin Li (Fellow, IEEE) received the BS degree in computer science from Tsinghua University, in 1985, and the PhD degree in computer science from the University of Houston, in 1990. He is a SUNY distinguished professor with the State University of New York. He has authored or co-authored more than 1130 journal articles, book chapters, and refereed conference papers. He holds nearly 80 patents announced or authorized by the Chinese National Intellectual Property Administration. He is an AAAS fellow, an AAIA fellow, an ACIS fellow, and an AIIA fellow. He is a member of the European Academy of Sciences and Arts. He is a member of Academia Europaea (MAE), a fellow of the Asia Computational Intelligence Society (ACIS), and a member of the SUNY Distinguished Academy.